基于代价敏感集成极限学习机的文本分类方法*

李明[1,2]　　肖培伦[2,3]　　张矩[1]　　顾心盟[4]

[1]（中国科学院重庆绿色智能技术研究院 高性能计算应用研究中心 重庆 400714）

[2]（清华大学信息技术研究院 语音与语言研究中心　北京 100084）

[3]（爱丁堡大学科学与工程科学学院　英国　EH1）

[4]（北京邮电大学　北京 100876）

摘要： 加权极限学习机对不同类别的样本赋予不同的权值，在一定程度上提高了分类准确率,但加权极限学习机只考虑了不同类别样本之间差异，忽视了样本噪声和同类样本之间的差异。本文提出了一种基于文本类别信息熵的极限学习机集成方法，该方法以 Adaboost.M1 为算法框架，通过文本的类内分布熵和类间分布熵生成文本类别信息熵，由文本类别信息熵构造代价敏感矩阵，把代价敏感极限学习机集成到 Adaboost.M1 框架中。实验结果表明，该方法与其他类型的极限学习机相比较有更好的准确性和泛化性。

关键词： 极限学习机　集成学习　Adaboost.M1　文本分类　代价敏感

分类号： TP393

Ensemble Extreme Learning Machine based on Cost Sensitive for Text Classification

Li Ming[1,2]　Xiao Peilun[2,3]　Zhang Ju[1]　Gu Xinmeng[4]

[1](High performance computing application R&D Center, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing ,400714, China)

[2] (Center for Speech and Language Technology,Research Institute of Information Technology, Tsinghua University,Beijing, 100084, China)

[3] (The University of Edinburgh,College of Science & Engineering, Edinburgh EH1, England)

[4] (Beijing University of Posts and Telecommunications, Beijing,100876, China)

Abstract: The weight extreme learning machine improves the classification accuracy, which gives different weight to different samples, but the weight extreme learning machine only takes into account the differences between samples in the different categories and neglects the difference between samples in the same category and noise. In this paper, we propose a novel method about ensemble extreme learning machine based on text information entropy, which takes Adaboost.M1 as the algorithm

* 本文系国家自然科学基金（项目编号：No.61672488）、重庆市社会事业与民生保障科技创新专项（项目编号：No.cstc2015shms-ztzx10005）、陆军军医大学第一附属医院重大领域技术创新计划项目（项目编号：No.SWH2016ZDCX1008）的研究成果之一。

framework, generates text information entropy through the intra-class entropy and the inter-class distribution entropy of the text, constructs a cost sensitive matrix by using the text information entropy, integrates the cost sensitive ensemble extreme learning machine into the Adaboost.M1 framework. The experimental results show that the proposed method has better accuracy and generalization than other extreme learning machines.
Keywords: extreme learning machine ensemble learning Adaboost.M1 text classification Cost-sensitive

1 引 言

文本分类是指在给定分类体系下，使用计算机自动地标记文本类别的过程。文本分类作为文本挖掘的关键技术之一，广泛应用于信息检索、搜索引擎、问答系统、舆情分析、情感分析等领域。随着网络技术的高速发展，网页的数量成几何速度增长，高效而个性化的信息检索需要发展更精确更有效的文本分类技术。目前比较成熟的文本分类方法有：k 近邻（k-nearest neighbor, K-NN）[1-2]、朴素贝叶斯(Naive Bayes)[3]、决策树(Decision tree)[4]、最大熵(maximum entropy)[5-6]、支持向量机(support vector machine,SVM)[2-7]、神经网络(neural networks)[8-9]、模糊理论[10]等。

Huang 等人提出了一种新型的单隐层前馈神经网络--极限学习机（ELM）[11]，该算法输入层与隐含层之间的连接权值和阈值均随机产生，而且在模型训练过程中无需对参数进行调整， 只需对隐含层神经元个数进行设置，避免了基于梯度下降学习方法的许多问题,如陷入局部极小、收敛速度慢等问题[12]，和传统的神经网络相比， ELM 具有相同的全局逼近能力 准确率高并且模型简单。与其他传统机器学习方法相比极限学习机也具有明显的优势,极限学习机具有更快的学习速度和更好的泛化性能。Ying Liu 等人对 ELM 和 SVM 在文本分类上的性能进行了对比[13]；Wenbin Zheng 等人使用潜在语义分析对文本进行降维，将正则化极限学习机（RELM）、神经网络和 SVM 对文本进行分类比较，RELM 表现出更快的学习速度和更好的分类性能[14]；此后 Wenbin Zheng 等人又提出了基于非负矩阵分解的线性分类器和基于 ELM 分类器想结合的文本快速分类框架[15]；Xiang guo Zhao 等人提出了一个基于极限学习机的 XML 文档分类框架，在该框架中对 voting-ELM 进行了改进，成功地将 REV 和 RCC 方法应用于 v-ELM，取得了比 voting-ELM 更好的效果[16]；此后，Xiang guo Zhao 继续对该方法进行改进，在 RCC 方法中引入 ε 参数提升重投的精确率，对 ELM 的投票结果进行概率统计，从而进一步提升了分类效果[17]；Lijuan Duan 采用 KELM 对历史专利文献进行分类，相对 SVM 取得了更好的效果[18]；于海燕等人通过信息增益对文本特征进行降维，引入小波核应用 KELM 对中文文本进行情感分类[19]；李永强通过优化搜索策略，提出了 CPSO-ELM 算法来选择单隐层前馈神经网络中隐藏节点的输入权重和偏置，对 XML 文档进行分类[20]；Rajendra Kumar Roul 等人研究了特征提取技术对 ELM 分类性能的提升，对单一 ELM 和多层 ELM 在文本分类领域做了大量的实验，结果超过了许多 state-of-the-art 的方法，包括 SVM 方法[21]；此后，Rajendra Kumar Roul 提出了一种基于 K 均值聚类的文本分类特征选择算法，结合 Wordnet 降低文本的维度，应用于单一 ELLM 和多层 ELM[22]。

然而，ELM 算法也存在着一定的不足，其预测性能仍受连接权值、阈值及隐含层节点数的影响[23]，由于其输入连接权值和隐含层阈值为随机初始化，在对相同训练样本及检验样本下多次执行此算法时的结果可能会有一定出入，即模型稳定性不理想。集成学习相比单个模型可以有效的提高预测性能[24]。Adaboost 是一种重要的集成学习技术，能够将预测精度仅比随机猜度略高的弱学习器增强为预测精度高的强学习器。Yunliang Jiang 等人通过 PCA 对人脸特征进行降维，将 ELM 嵌入 Adaboost 框架中应用于人脸识别[25]。黄海波等人提出基于 Adaboost 的 ELM 算法，通过小波包分解对减振器异响特征信息进行提取，对减振器异响声品质进行预[26]。Yan Xu 等人将基于 Adaboost 的 ELM 方法应用于交通标志的识别[27]。Kuan Li 等人提出了一种 boosting 的加权 ELM 方法，将加权 ELM 无缝嵌入到 boosting 的 ELM 中，在 Adaboost 每次迭代中调整样本的分布权重。但该方法只考虑了数据集的类间不平衡，而没有考虑类内的不平衡，实际上，类内的不平衡对分类性能的影响也很大[28]。

本文在 Adaboost.M1 的框架下结合代价敏感 ELM 提出了一种新的 ELM 模型并将其应用到文本分类中。首先使用词向量得到高质量低维度的文本特征向量，然后用文本类别信息熵构建代价敏感矩阵，把代价敏感加权 ELM 作为基分类器，在多分类 Adaboost.M1 框架中通过代价敏感因子调整样本分布。最后，在三个文本标准数据集 20newsgroups、Reuters52 和 Webkb 上对比了 ELM、Voting-ELM、Adboost-WELM 以及本文提出的 Adaboost-WELM 框架下 AE1-WELM、AE2W-ELM 和 AE3W-ELM 三种方法的精度和泛化性能。通过实验验证，AE3-WELM 算法相比其他的 ELM 方法有着更为显著的分类性能、稳定性和泛化性。

2 基于 Adaboost 的加权极限学习机

2.1 加权极限学习机

加权极限学习机(weight extreme learning machine)是在 ELM 的基础上引入加权矩阵 W，对每一个样本进行加权，减少样本类间可能存在的不平衡性，从而提高样本总体的识别率。根据 KKT 理论有：

$$\beta = H^{+}T = \begin{cases} \left(\frac{I}{C} + H^{T}WH\right)^{-1} H^{T}T, & N \ge L \\ H^{T}\left(\frac{I}{C} + WHH^{T}\right)^{-1} T, & N < L \end{cases} \tag{1}$$

其中 W 为对角矩阵，对角线上的每一个元素为样本的权重值。W. Zong 等人经验地给出了两种加权方案[28]：

$$\text{方案 1：} W_1 = \frac{1}{\#t_i} \tag{2}$$

$$\text{方案 2：} W_2 = \begin{cases} \frac{0.618}{\#t_i}, & \#t_i > AVG(\#t_i) \\ \frac{1}{\#t_i}, & \#t_i \le AVG(\#t_i) \end{cases} \tag{3}$$

但加权极限学习机只是简单的用大类的样本数和小类的样本数来赋予样本权值，对少数类样本赋予更大的权重，但是同类样本的分配权重是相等的，这样只是考虑了数据集的类间不平衡，而没有考虑类内的不平衡，实际上，类内的不

平衡对分类性能的影响也很大。Kuan Li 等人在此基础上进行了改进，对于不同类样本权值采用不同更新方式，但是同样没有考虑同类样本之间的权值差异[28]。

2.2 基于 Adaboost 的加权极限学习机

AdaBoost 算法基本思想是将若干个弱分类器按照某种规则组合起来，集成为一个分类能力很强的强分类器。Freund 和 Schapire 改进了原本用于二分类问题的 Adaboost，生成 Adaboost.M1、AdaboostM2 算法用于多分类问题，同时给出了 Adaboost.M1 的扩展形式，本文使用 Adaboost.M1 算法的扩展形式[29]，然后将 1.1 中的加权极限学习机嵌入到 Adaboost.M1 框架中，生成基于 Adaboost 的加权极限学习机算法， 算法中采用 $h_m(x)$ 表示第 m 个弱分类器，通过弱分类器权重 α_m 对弱分类器进行组合得到强分类器，算法的主要步骤如下：

步骤 1 用式（2）初始化样本权值， $D_1(x_i)=1/\#t_k$ ， $i=1,2,\ldots,N$ ；

$\#t_k$ 表示样本所在类 t_k 所含样本个数；

步骤 2 对样本权值归一化 $D_1(x_i)=D_1(x_i)\big/\sum_{i=1}^{n}D_1(x_i)$ ；

步骤 3

For $m=1:M$ （M 为弱分类器数量）

(1) 用弱学习算法训练样本得到弱分类器 $h_m(x)$；

(2) 计算 $h_m(x)$ 分类错误率

$$\varepsilon_m=\sum_{i=1}^{N}D_m(x_i)I\left(h_m(x_i)\neq y_i\right) \tag{4}$$

(3) 当分类器分类错误率大于 0 而且小于 0.5 时，按照式（5）更新样本权值，否则退出循环；

$$D_{m+1}(x_i)=\frac{D_m(x_i)\exp\left(-\alpha_m I\left(h_m(x_i)\neq y_i\right)\right)}{Z_m} \tag{5}$$

Z_m 为归一化因子， $Z_m=\sum_{i=1}^{N}D_{m+1}(x_i)$ (6)

$$\alpha_m=\log\frac{1-\varepsilon_m}{\varepsilon_m}+\log(k-1) \tag{7}$$

步骤 4 组合分类器输出为：

$$\Theta(x)=\arg\max_k\sum_{m=1}^{M}\alpha_m I\left(h_m(x)=k\right)$$

3 基于代价敏感集成极限学习机的文本分类方法

3.1 基于词向量的文档向量生成

数量巨大的训练样本和过高的向量维度是文本分类的特点。过高维度的特征集会增加极限学习机的计算负担。传统的做法是降低文本向量空间的维数并减少

噪音信息对文本分类的干扰，保证文本分类的精度。词向量文本表示比传统的人工提取特征向量的方法具有更好的特征表达效果，词向量通过训练无标注语料将每个词映射成低维实数向量[30]，通过低维实数向量之间的距离来描述词语之间的语义相似度，同时又能有效避免特征向量的维度灾难。Mikolov 等人提出了两种词向量学习模型： CBOW（Continuous Bag of Words）和 Skip-gram 模型[31]。Skip-gram 模型以当前词作为对数线性分类器的输入，预测上下文中的词语。

给定词序列 $w = \{w_1, w_2, \cdots w_N\}$， N 为序列长度，在 skip-gram 的 NN 网络结构中，输入词序列中的第 i 个词 w_i，使用当前词 w_i 预测窗口大小为 b 的上下文， Skip-gram 模型最大化的目标函数如式（8）所示：

$$\frac{1}{N}\sum_{i=1}^{N}\sum_{-b\le j\le b, j\ne 0}\log p\left(w_{i+j} \mid w_i\right) \tag{8}$$

采用 softmax 函数计算 Skip-gram 模型定义的 $p\left(w_{i+j} \mid w_i\right)$ 如式（9）所示:

$$p\left(w_{i+j} \mid w_j\right) = \frac{\exp\left(c_{w_{i+j}} c_{w_j}\right)}{\sum_w \exp\left(c_w c_{w_j}\right)} \tag{9}$$

其中， $c_{w_{i+j}}$ 和 c_{w_j} 分别为 w_{i+j} 和 w_j 的词向量。

Skip-gram 模型在文本相似性度量和文本分类任务上都有较好的表现，本文采用的词向量模型为 Skip-gram 模型。我们首先产生特征词的词向量 $c_{w_i} = (v_1, v_2, \cdots v_m)$， 表 m 示词向量的维度，我们用 $c_{i,j}$ 表示第 i 个文档中第 j 个单词的词向量，通过式（10）生成文档向量：

$$v_i = (1/J_i)\sum_{j=1}^{J_i} c_{i,j} \tag{10}$$

其中 J_i 表示第 i 个文档中单词的个数。

3.2 类别信息熵

通常为了提高文本分类的性能，研究人员主要从两个方面开展研究：一是改善分类算法（或学习模型）；二是改善文本数据表示模型。传统上，我们通过向量空间模型（vector space model，VSM）来表示文本[32]，就是在分类之前把每个文本文档都表示成由一定数量的特征词的权重值所组成的向量。特征词在不同类别的文本中出现具有一定的不确定性，这种不确定性可用熵（entropy）来度量。特征词的权重应当根据它在文本分类中的重要性来分配，而特征词的重要性体现在它的类别区分力的大小，因为类别区分力大的词有助于区分不同类别的文本。我们采用香农的信息熵来度量特征词区分类别的能力，得到特征词的类间信息熵函数的定义。

定义 1：若训练集文本有 m 个类别，特征词 t_i 在类别 c_j ($j=1,2,\ldots,m$) 的文档中出现的频率为 DF_{ij}，在所有的文档中出现的频率为 DF_i，则特征词 t_i 的类间信息熵函数（Entropy with Difference 记为 $ED(t_i)$ ），定义为：

$$ED(t_i)=\ln\left(\frac{1}{E_d(t_i)+\theta}\right) \tag{11}$$

其中，$E_d(t_i)$ 为特征词 t_i 的类间信息熵，$E_d(t_i)=-\sum_{k=1}^{m}\left(\frac{DF_{ij}}{DF_i}\right)\times\ln\left(\frac{DF_{ij}}{DF_i}\right)$，$DF_i=\sum_{j=1}^{m}DF_{ij}$。由信息熵的定义以及熵最大值定理可知，特征词 t_i 在各个类别中分布得越均匀，熵值 $E_d(t_i)$ 越大。特征词 t_i 在各个类别中分布得越不均匀，熵值 $E_d(t_i)$ 越小。当且仅当特征词 t_i 在各类别中均匀分布时，熵值 $E_d(t_i)$ 最大。所以，我们对 $E_d(t_i)$ 取倒数，为了防止分母为零，我们加上一个 θ 参数。通过实验我们发现文档的类间分布熵取完倒数后的数值 $1/(E_d(t_i)+\theta)$ 通常较大，虽然文档之间的类间信息熵差异很大，可以起到刻画文档分布的作用，但是淹没了其后的类内信息熵的数值，使得类内信息熵对文档分布的作用完全不能显现，我们对 $1/(E_d(t_i)+\theta)$ 取对数，最终得到文本的类间信息熵。

定义 2：若训练集文本有 m 个类别，特征词 t_i 在类别 c_j ($j=1,2,\ldots,m$) 的第 k 个文档中出现的频率为 $TF(t_i,d_{jk})$，则特征词 t_i 的类内信息熵函数（Entropy with Contribution，记为 $EC(t_i)$ ），定义为：

$$EC(t_i)=e^{\mathop{Max}\limits_{j\in[1,m]}\left(E_c(t_i,c_j)\right)} \tag{12}$$

其中，$E_c(t_i,c_j)=-\sum_{j=1}^{|c_j|}\frac{TF(t_i,d_{jk})}{TF(t_i,c_j)}\ln\left(\frac{TF(t_i,d_{jk})}{TF(t_i,c_j)}\right)$，$E_c(t_i,c_j)$ 为特征词 t_i 对类别 c_j 的类内信息熵，$|c_j|$ 表示 c_j 类中的文本数量，d_{jk} 表示第 c_j 类中第 k 个文档；$TF(t_i,d_{jk})$ 表示特征词 t_i 在第 c_j 类中第 k 个文档 d_{jk} 出现的频率，$TF(t_i,c_j)$ 表示特征 t_i 在 c_j 类文本中出现的总频率。对于类别 c_j ($j=1,2,\ldots,m$)，我们取 $E_c(t_i,c_j)$ 中的最大值作为特征词 t_i 的类内信息熵。

考虑到有些具有较高类别区分能力的特征词是低频词，而低频词的类内信息熵比较低，甚至几乎为 0，如果直接用 $\underset{j\in[1,m]}{Max}\left(E_c\left(t_i,c_j\right)\right)$ 作为类内信息熵函数，会导致该特征词的类间信息熵和类内信息熵结合之后，得到错误的信息度量结果；而且通过实验我们发现加上 e 作为调节因子后的特征信息熵函数比直接使用 $\underset{j\in[1,m]}{Max}\left(E_c\left(t_i,c_j\right)\right)$ 的特征信息熵函数有更好更显著的文本区分能力。结合特征词的类间信息熵函数和类内信息熵得到特征词的类别信息熵函数：

$$EDC\left(t_i\right)=ED\left(t_i\right)\times EC\left(t_i\right) \tag{13}$$

特征词的类别信息熵值越大，说明该特征词区分类别的作用越明显，对文本分类而言，该特征词具有很高的区分度和重要度，通过特征词的类别信息熵对不仅刻画了不同类别之间的文本分布，同时刻画了同一类别内部的文本分布，对文本的描述具有比较细的粒度。文档类别信息熵生成算法的主要步骤如下：

步骤 1 对文本集 $D=\{d_1,d_2,\dots,d_n\}$ 中的文本 $d_i\ (1\le i\le n)$ 进行预处理；

步骤 2 预处理后，文本集中每篇文本 d_i 被表示成特征词的集合 $d_i=\left\{t_{i1},t_{i2},\dots,t_{i|d_i|}\right\}$；

步骤 3 计算训练文档中每个特征词的类间信息熵

For $i=1:|d_n|$ （$|d_n|$ 为 d_i 文档中特征词的数量）

计算文档中特征词 t_i 的 DF_i 值和 DF_{ij} 值；

根据式（12）计算文档中特征词 t_i 的 $ED\left(t_i\right)$ 值；

步骤 4 计算训练文档中每个特征词的类内信息熵

For $j=1:m$

For $k=1:|c_j|$ （$|c_j|$ 为第 c_j 类中文档的数量）

计算文档中特征词 t_i 的 $TF\left(t_i,d_{jk}\right)$ 值、$TF\left(t_i,c_j\right)$ 值和 $E_c\left(t_i,c_j\right)$ 值；

根据式（14）计算文档中特征词 t_i 的 $E_c\left(t_i\right)$ 值；

步骤 5 根据式（13），计算文档中特征词 t_i 的 $EDC\left(t_i\right)$ 值；

步骤 6 对 $EDC\left(t_i\right)$ 值归一化；

步骤 7 计算文本集中每个文档的 EDC 值：$EDC=\sum_{i=1}^{|d_n|}EDC\left(t_i\right)$。

3.3 基于代价敏感的集成极限学习机

加权极限学习机和 Boosting 的加权极限学习机都没有考虑同类样本之间的权值差异，为了进一步提高极限学习机的分类性能，我们将代价敏感引入到极限学习机中，为不同的样本赋予不同的权重 $w_i\ (i=1,2,\dots,n)$，我们用文本的类别

信息熵构建代价敏感矩阵 $W = \begin{bmatrix} w_1 & & \\ & \ddots & \\ & & w_n \end{bmatrix}$，其中 $w_i = EDC(x_i)$，代价敏感加权极限学习机的输出参数 β 为：$\beta = (WH)^{+} WT$ 。

Adaboost.M1 算法是通过对训练样本的自适应采样，调整样本权重来调整样本分布，对于错误分类的样本分配更大的权值，对于正确分类的样本赋予更小的权重，这样的权重分配是从错分率上体现每个样本的重要性，实际上同类别样本集中不同样本之间的重要性也有着很大的差异。我们利用 Adaboost.M1 算法在迭代中权重分配的思想，把代价敏感因子引入 Adaboost.M1 框架中，将加权极限学习机无缝集成到代价敏感 Adaboost.M1 框架中。我们通过文本的类别信息熵来刻画每个样本对于类别区分的重要程度，然后通过文本类别信息熵来构建代价敏感因子，在每次迭代中根据每个样本的重要性来更新样本权重。根据训练样本权重更新方法的不同，分别记为 AE1-WELM、AE2-WELM 和 AE3-WELM，本文将这三种方法统称为 AEx-WELM 方法，算法的主要步骤为：

步骤 1 对训练数据集和测试数据集进行预处理，去除停用词、去除特殊符号，文本集中每篇文档 d_i 被表示成特征词的集合 $d_i = \{t_{i1}, t_{i2}, \ldots, t_{i|d_i|}\}$；

步骤 2 利用 Word2vec 生成特征词的词向量；

步骤 3 每篇文档 d_i 用文档向量 $v_i = (1/J_i)\sum_{j=1}^{J_i} c_{i,j}$ 表示；

步骤 4 生成代价敏感矩阵 $W_t = diag(EDC(x_i)), i = 1, \ldots, N$；

步骤 5 训练权重为 W_t 的加权 ELM，并作为弱分类器 $h_t(x)$；

步骤 6 初始化训练文档权值分布 $D_1(x_i) = EDC(x_i)$，$i = 1, \ldots, n$

步骤 7 For $t = 1 : T$ (T 为弱 分类器数量)

(1) 计算 $h_t(x)$ 分类错误率：同式（4）；

(2) 当分类器分类错误率大于 0 而且小于 0.5 时，按照式（14）更新样本权值，否则退出循环；

$$D_{t+1}(x_i) = D_t(x_i) \times \begin{cases} e^{EDC(x_i)\alpha_t} & \text{(14a)} \\ EDC(x_i)e^{\alpha_t} & \text{(14b)} \\ EDC(x_i)e^{EDC(x_i)\alpha_t} & \text{(14c)} \end{cases}$$

$D_t(x_i)$ 值的归一化同式（6），α_t 值同式（7）；

步骤 8 给定测试样本 x，输出测试样本的类别标签

$$\Theta(x) = \arg\max_k \sum_{t=1}^{T} \alpha_t [h_t(x) = k]$$

AE2-WELM 和 AE3-WELM 算法和 AC1-WELM 算法基本一致，区别在于样本权值的更新上。用式（14a）更新样本权值的为 AE1-WELM 算法，用式（14b）

更新样本权值的为 AE2-WELM 算法，用式（14c）更新样本权值的为 AE3-WELM 算法。

4 实验

4.1 实验数据集

我们将所有的模型在三个标准文本数据集上进行实验。由 Ken Lang 收集的 20 Newsgroups 数据集[1]、由 CMU 项目收集的 webkb 数据集[2]、由 DavidD Lewis 发布的 Reuters52 数据集[3]，第一个是平衡数据集，后两个是非平衡数据集。20Newsgroups 语料库包含 20 个不同类别的英文新闻，其中总文档数为 18846 个，为了提高实验的可靠性，所有的重复文件被删除，剩下 11293 个文档被用作训练数据集和 7528 个文档被用作训练数据集。原始 WebKB 的语料库包含约 8300 个英文网站，分为 7 类，我们选择最常用的 4 大类，包括 student、faculty、course 和 project 4 个文本子集，共有 4199 个文档。同样，为了提高实验的可靠性，重复的文件被删除，剩下 2756 文档被用作训练数据集和 1375 个文档被用作测试数据集。Reuters52 数据集中 90 类中最常使用的 52 类称为 R52 数据集。R52 数据集总共 9100 个文档，其中 6532 个文档被用作训练数据集，2568 个文档被用作测试数据集。我们对数据集首先进行预处理，包括：删除停用词、去掉单个字符和非字母符号、把大写字母转化成小写字母、词干还原、去除低频词。我们使用 google 提供的词向量训练工具 word2vec 进行词向量模型训练[4]。

4.2 评价指标

精确率（Precision）、查全率（Recall）和 F1 值被广泛应用于分类效果评价。微平均和宏平均是两种对分类结果进行全局评价的方法：微平均（Micro-average）是先计算所有文档的分类结果，然后对所有文档求平均；宏平均（Macro-average）是先计算各个类别的分类结果，再对所有类别求平均。具体定义如下：

$$\text{微平均}\ MicroF1 = \frac{2 \times MicroP \times MicroR}{MicroP + MicroR},$$

$$\text{宏平均}\ MacroF1 = \frac{2 \times MacroP \times MacroR}{MacroP + MacroR},$$

其中 $MicroP = \left(\sum_{i=1}^{m} a_i\right) \Big/ \sum_{i=1}^{m} b_i$；$MicroR = \left(\sum_{i=1}^{m} a_i\right) \Big/ \sum_{i=1}^{m} d_i$；$MicroP$ 和 $MicroR$ 分别表示微平均的精确率和查全率；$MacroP$ 和 $MacroR$ 分别表示宏平均的精确率和查全率；b_i 是测试集中 c_i 类的文档数；a_i 是其中被正确判断为 c_i 类的文档数；d_i 是属于 c_i 类的文档数。微平均倾向于大类，宏平均倾向于小类。为了对分类的整体性能有一个度量，本文采用 $F1$ 的微平均 $MicroF1$ 和宏平均 $MacroF1$ 、训练时间(Training time)和测试时间(Testing Time)对分类结果进行评价。

1 http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo20/www/data/news20.html

2 http://web.ist.ult.pt/~acardoso

3 https://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

4 https://code.google.com/p/word2ve

4.3　实验结果分析

（1）参数设置

所有实验均运行在 2.4GHZCPU 和 4G 内存环境下。ELM 相关算法由 python 语言实现，每个实验运行 10 次，取平均值作为结果。文本输入维数在 50 维到 500 维之间进行取值。极限学习机的激活函数通常有：Sigmoid、RBF 和 tanh 函数，通过对以上激活函数性能的比较，我们选取 tanh 函数作为隐藏节点的激活函数：极限学习机的超参数 c 和 L 采用网格搜索法进行选取，$\{10^{0},10^{-1},...,10^{-8}\}$ 为 c 值的搜索范围，L 的搜索范围{100,200,...1000}。

（2）性能比较

为了验证 AEx-WELM 的性能，我们把 AEx-WELM 和 ELM、Voting-ELM 和 Adaboost-WELM 在三个标准数据集上做了比较。为了检测这些方法在文本维数变化情况下的性能，我们选取了不同的文本维数，将微平均 mf1 、宏平均 MF1、训练时间和测试时间的变化情况作比较，具体结果见表 1-3，表中数据均

表 1 在 20newsgroups 上分类性能对比

dim	Evaluation measures	ELM	Voting-ELM	Adaboost-WELM	AE1-WELM	AE2-WELM	AE3-WELM
50	mf1	0.756	0.768	0.747	0.772	0.749	0.773
	MF1	0.743	0.754	0.741	0.764	0.749	0.764
	Training time(s)	5.744	114.608	243.455	242.735	261.990	262.045
	Testing time(s)	1.336	27.125	31.078	31.750	31.416	31.368
100	mf1	0.773	0.786	0.771	0.790	0.771	0.791
	MF1	0.759	0.772	0.764	0.781	0.763	0.782
	Training time(s)	6.639	140.346	244.671	244.628	244.818	244.773
	Testing time(s)	1.373	28.121	31.702	31.626	31.687	31.639
200	mf1	0.784	0.800	0.792	0.804	0.791	0.806
	MF1	0.770	0.786	0.783	0.794	0.783	0.796
	Training time(s)	5.931	119.746	247.969	248.189	248.280	248.606
	Testing time(s)	1.396	28.462	33.048	32.989	33.051	32.977
300	mf1	0.785	0.802	0.791	0.804	0.792	0.805
	MF1	0.771	0.787	0.782	0.795	0.784	0.796
	Training time(s)	6.037	122.124	306.810	328.235	322.786	303.106
	Testing time(s)	1.446	29.383	35.506	35.652	35.759	35.705
400	mf1	0.788	0.805	0.795	0.809	0.780	0.808

	MF1	0.774	0.790	0.786	0.800	0.772	0.799
	Training time(s)	6.122	123.236	254.772	255.044	254.767	254.918
	Testing time(s)	1.506	30.449	35.167	35.082	35.241	35.090
500	mf1	0.786	0.805	0.797	0.808	0.795	0.809
	MF1	0.773	0.790	0.788	0.799	0.786	0.800
	Training time(s)	7.790	157.031	324.748	311.323	313.438	307.131
	Testing time(s)	1.631	33.289	37.824	38.742	38.265	38.377

表 2 在 Reuters52 上分类性能对比

dim	Evaluation measures	ELM	Voting-ELM	Adaboost-WELM	AE1-WELM	AE2-WELM	AE3-WELM
50	mf1	0.917	0.919	0.920	0.931	0.920	0.929
	MF1	0.607	0.614	0.644	0.661	0.620	0.660
	Training time(s)	4.828	96.317	110.873	110.953	111.173	111.093
	Testing time(s)	0.354	7.240	8.597	8.580	8.593	8.590
100	mf1	0.918	0.923	0.927	0.938	0.925	0.936
	MF1	0.602	0.615	0.661	0.684	0.666	0.671
	Training time(s)	4.842	102.547	111.240	111.207	111.303	111.207
	Testing time(s)	0.366	7.900	9.187	9.127	9.023	9.127
200	mf1	0.922	0.925	0.925	0.938	0.926	0.937
	MF1	0.621	0.616	0.650	0.682	0.661	0.679
	Training time(s)	4.880	98.303	114.353	114.343	114.290	114.360
	Testing time(s)	0.390	7.850	9.710	9.673	9.695	9.660
300	mf1	0.923	0.925	0.926	0.938	0.926	0.936
	MF1	0.618	0.626	0.656	0.674	0.662	0.678
	Training time(s)	4.990	100.000	112.283	112.460	112.573	112.683
	Testing time(s)	0.410	8.138	9.578	9.578	9.583	9.560
400	mf1	0.922	0.926	0.925	0.939	0.927	0.939
	MF1	0.617	0.618	0.659	0.685	0.653	0.680
	Training	5.002	101.698	118.338	118.345	118.425	118.410

	time(s)						
	Testing time(s)	0.422	8.498	10.530	10.483	10.473	10.513
500	mf1	0.923	0.925	0.928	0.937	0.926	0.937
	MF1	0.621	0.625	0.666	0.682	0.665	0.679
	Training time(s)	5.190	102.505	118.398	118.355	118.428	118.575
	Testing time(s)	0.438	8.883	10.423	10.400	10.423	10.418

表 3 在 Webkb 上分类性能对比

dim	Evaluation measures	ELM	Voting-ELM	Adaboost-WELM	AE1-WELM	AE2-WELM	AE3-WELM
50	mf1	0.850	0.873	0.864	0.876	0.867	0.874
	MF1	0.834	0.860	0.850	0.865	0.851	0.857
	Training time(s)	2.078	41.602	69.995	70.121	70.059	70.237
	Testing time(s)	0.276	5.616	6.935	6.896	70.059	6.890
100	mf1	0.863	0.888	0.881	0.886	0.877	0.885
	MF1	0.850	0.876	0.867	0.879	0.863	0.874
	Training time(s)	2.084	41.696	70.215	70.387	70.376	70.409
	Testing time(s)	0.281	5.684	7.041	6.954	7.003	6.951
200	mf1	0.866	0.896	0.883	0.892	0.881	0.894
	MF1	0.853	0.884	0.869	0.884	0.867	0.885
	Training time(s)	2.129	42.614	71.397	71.572	71.467	71.638
	Testing time(s)	0.288	5.877	7.188	7.149	7.173	7.152
300	mf1	0.866	0.897	0.884	0.893	0.884	0.893
	MF1	0.854	0.886	0.871	0.884	0.871	0.885
	Training time(s)	2.165	43.321	72.933	73.062	73.008	73.153
	Testing time(s)	0.322	6.482	7.777	7.726	7.744	7.726
400	mf1	0.865	0.893	0.884	0.893	0.880	0.893
	MF1	0.851	0.882	0.871	0.883	0.865	0.883
	Training time(s)	2.211	44.316	73.760	73.795	73.799	74.188
	Testing	0.306	6.190	7.477	7.449	7.505	7.445

	time(s)						
500	mf1	0.868	0.894	0.882	0.890	0.881	0.892
	MF1	0.853	0.881	0.882	0.880	0.867	0.882
	Training time(s)	2.279	45.458	74.773	74.970	74.876	75.059
	Testing time(s)	0.314	6.399	7.669	7.629	7.663	7.626

是各个方法在最佳（c,L）取值下的最优性能值。从表 1-3 中可以看出，在所有测试中 ELM 表现都是最差的。在三个标准数据集上，AE1-WELM 和 AE3-WELM 的 MF1 值高于其他所有 ELM 方法；在 20newsgroups 数据集和 Reuters52 数据集上，AE1-WELM 和 AE3-WELM 的 mf1 值明显高于其他 ELM 方法,比其他所有的 ELM 方法有更好的性能，说明把代价敏感极限学习机结合到 Adaboost 框架中是有效的。在非平衡数据集 Reuters5252 和 WEbkb 上，AE1-WELM 和 AE3-WELM 的性能明显高于 ELM、Adaboost-WELM 和 AE2-WELM，说明文本提出的方法可以改善不平衡多类分类问题的分类效果；同时在平衡数据集 20newsgroups 上，AE1-WELM 和 AE3-WELM 性能提升就更为明显，超过了其他所有的 ELM 方法；而且在三个文本标准数据集上，AE1-WELM 和 AE3-WELM 都表现稳定，说明这两种方法很好的泛化性能。在 AEx-WELM 三种方法中，AE2-WELM 表现最差，几乎和 Adaboost-WELM 的性能差不多，AE2-WELM 分布权重在迭代过程中变化剧烈，这可能导致 AE2-WELM 分类效果没有另外两种好的原因。AE1-WELM 和 AE3-WELM 两者之间相比，后者随着文本维数的增加展现出更优的分类性能和稳定性，所以 AE3-WELM 是文本推荐的方法。

AEx-WELM 和 ELM 相比：从图 1-3 中可以看到，对于所有的数据集，将极限学习机嵌入 Adaboost 框架中的四种方法 Adaboost-WELM、AE1-WELM、AE2-WELM、AE3-WELM 都比 ELM 方法在分类性能上有显著提高，其中 AE1

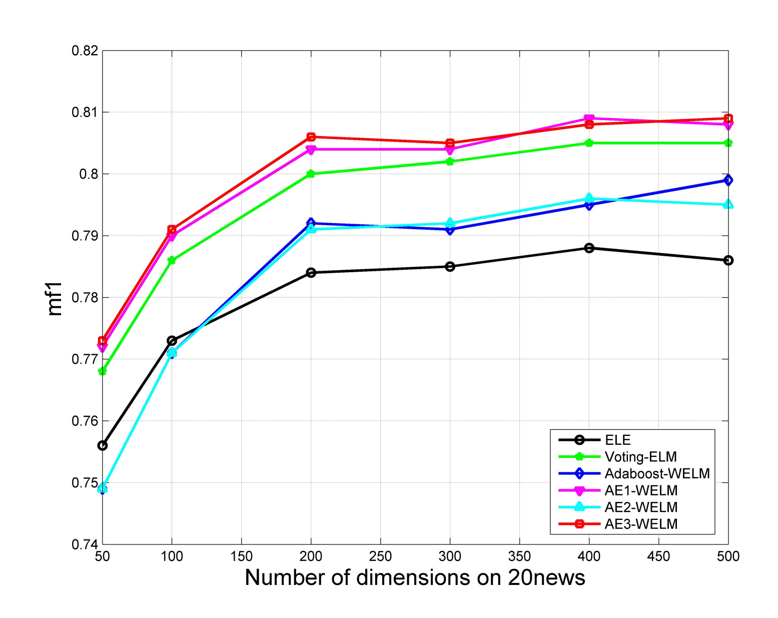


图 1　20newsgroups 数据集上的分类性能

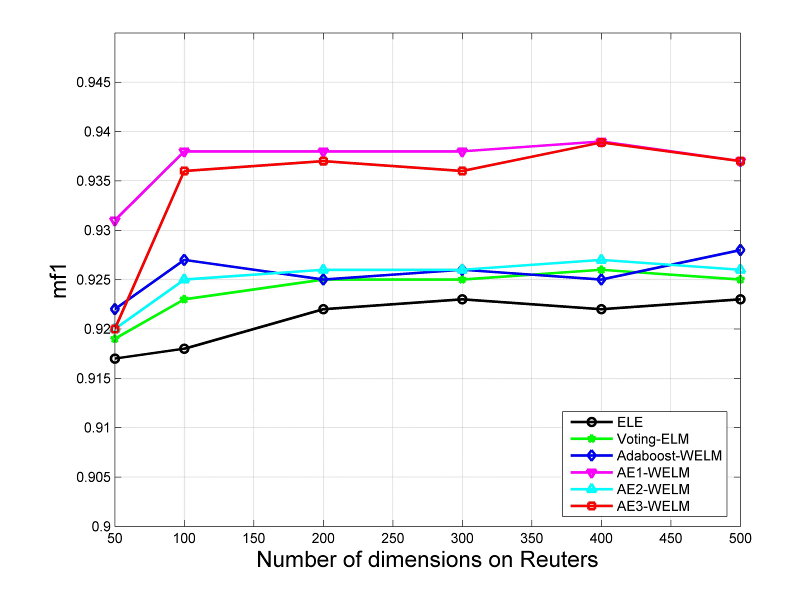


图 2　Reuters52 数据集上的分类性能

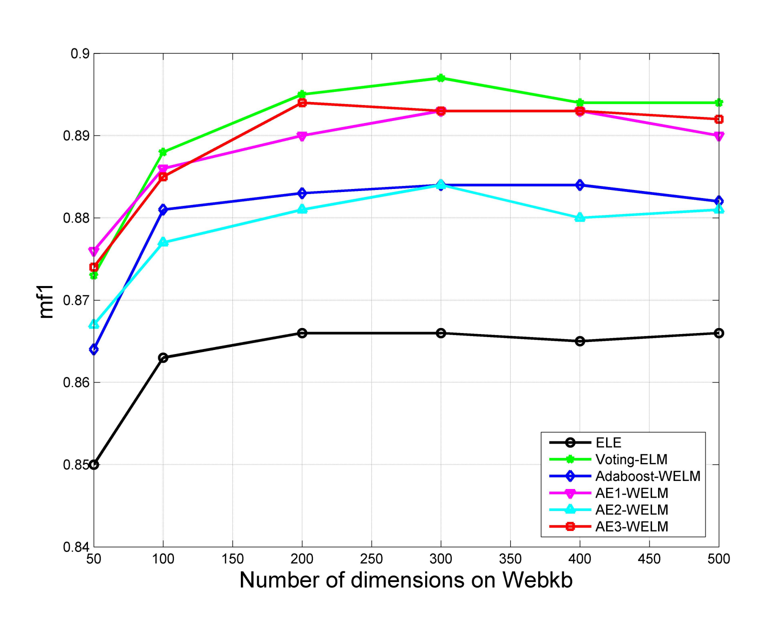


图 3　Webkb 数据集上的分类性能

-WELM 和 AE3-WELM 有着更为明显的优势。在三个标准数据集中，AE1-WELM 和 AE3-WELM 的综合指标 MF1 值都超过了 ELM 和其他类型的 ELM 的 MF1 值，尤其在 20newsgroups（平衡数据集）和 Reuters52（非平衡数据集）上的 MF1 值比起其他所有的 ELM 方法有着更为明显的提高；在 Webkb 数据集上（非平衡数据集）上的 mf1 值要略低于 Voting-ELM，但是 MF1 值仍然高于 Voting-ELM 方法。

AEx-WELM 和 Voting-ELM 比：从图 1-3 中可以看到，在平衡数据集 20newsgroups 上 AE1-ELM 和 AE3-WELM 比 Voting-ELm 要略微高一点，在非平衡数集 Reuters52 上 Voting-ELM 效果要比 AE1-ELM 和 AE3-WELM 差很多。但是在非平衡数据集 Webkb 上，Voting-ELM 却取得了比其他所有 ELM 都更好的效果，这是由于 Webkb 数据集较小，而且我们实验中只取了 Webkb 常用的 4

类，类的数目也较小，而 mf1 值是倾向于大类的，所以 AE1-ELM 和 AE3-WELM 的 mf1 值会比 Voting-ELM 略低，当然这也和 Voting-ELM 方法中我们采用的随机采样算法将数据集中文本特征的多样性充分发挥出来有一定的关系，所以在三个文本数据集中 Voting-ELM 都表现出了不错的性能；尽管如此在 Webkb 数据集上，AE1-ELM 和 AE3-WELM 的 MF1 值还是略微比 Voting-ELM 高出一点；而且到了像 20newsgroups 和 Reuters52 这类文本数量要远远多于 WEbkb 的数据集中，Voting-ELM 的性能就明显下降，不能取得像 AE1-ELM 和 AE3-WELM 同样显著的分类效果。

基于词向量的文本信息表达也一定程度上丰富了文本的特征表示，不仅有效降低了文本维度，而且在低维空间上（通常 100 维）就已经可以取得传统文本 VSM 特征表达在 1000 维（甚至更高维度）上的分类性能；同时我们从图 1-3 中也观察到，6 种的 ELM 方法在文本维数超过 400 维之后性能都呈现下降趋势，这说明过高的维数不仅给极限学习机造成了负担而且增加了噪音，而且影响了分类性能。

图 4-6 显示在三个文本标准数据集上，当文本向量为 300 维时，随着隐节点和 c 值变化的 AE3-WELM 算法 mf1 值的变化。从图中我们可以观察到，虽然较多的隐藏节点能够帮助 AE3-WELM 取得更好的分类结果，但是当隐藏节点数超过 400 以后其 mf1 值较为稳定，过大的隐藏节点数起不到什么太大影响。当隐藏节点达到一定程度后，分类性能会变得不稳定，超过 800 以后分类性能会性能随着隐藏节点的增加而下降，这种情况应该是由于过拟合造成的。从图 4-6 中可以看出，正则化参数 c 是一个很关键的因素，性能随着 c 值的减少达到一个稳定值。正则化参数 c 对性能的影响要大于隐藏结点，当 c 值较小时接近 10^{-5} 时，AE3-WELM 对于隐藏节点参数的选取并不敏感。

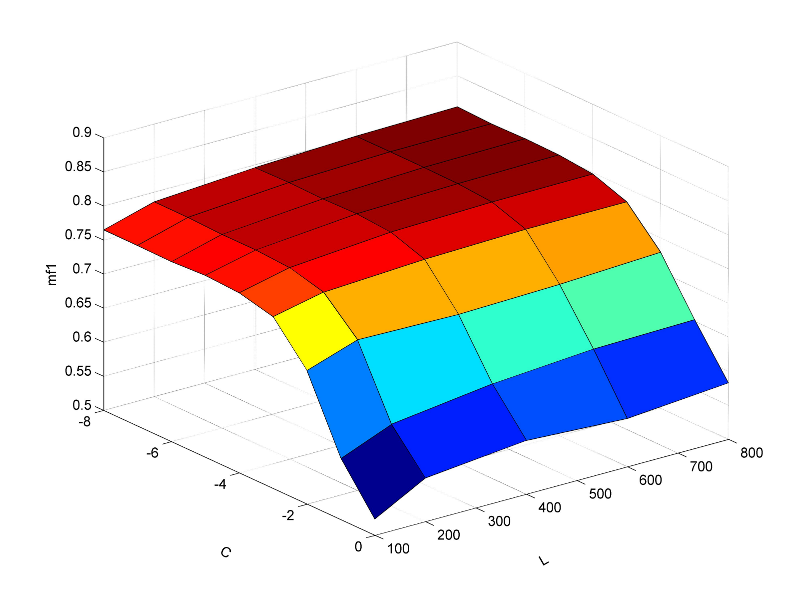


图 4　随着隐节点和 c 值变化的 AE3-WELM
mf1 值的变化（20newsgroups 数据集）

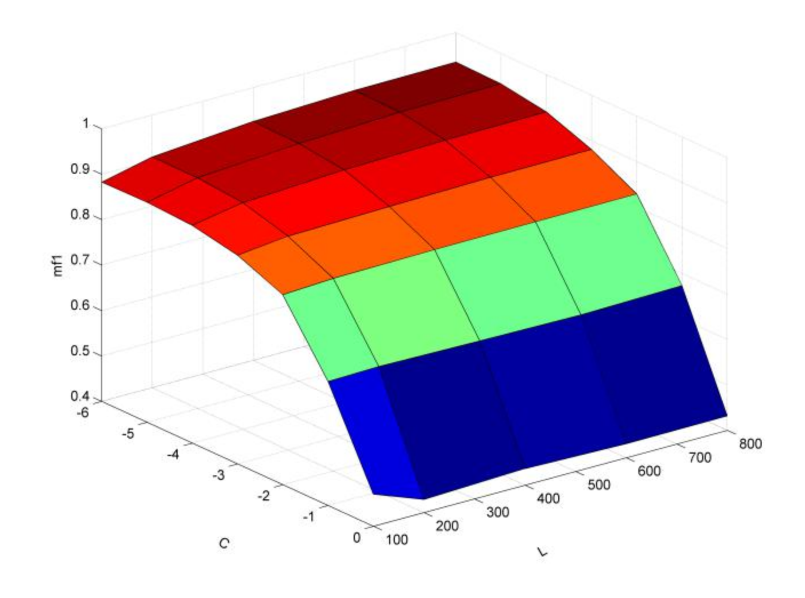


图 5　随着隐节点和 c 值变化的 AE3-WELM mf1 值的变化（Reuters52 数据集）

图 7-9 显示了三个文本标准数据集上，当文本向量为 300 维时，6 种 ELM 方法随着文本向量维度的变化模型训练时间的变化。因为 Voting-ELM 需要集成多个子分类器，而 Adaboost-WELM 和 AEx-WELM 方法都需要经过较多的迭代，所以它们的训练时间消耗要比 ELM 长很多。AEx-WELM 和 Adaboost-WELM 的训练时间要比 ELM 长很多，比 Voting-ELM 也要长出一些。但是我们更观察到

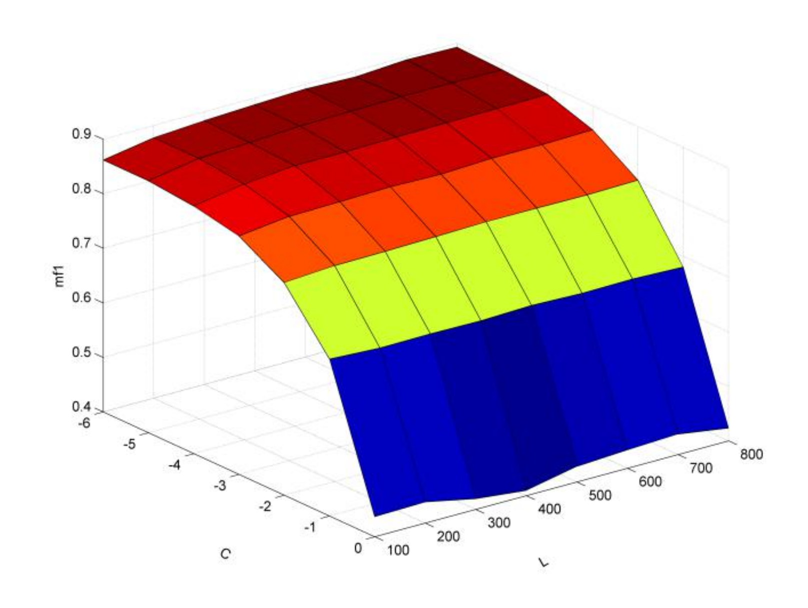


图 6 随着隐节点和 c 值变化的 AE3-WELM mf1 值的变化（Webkb 数据集）

这样的细节：Voting-ELM 的训练时间是 ELM 的倍数，这个倍数取决于 Voting-ELM 当中使用的集成分类器数量，所以这个训练时间的增加是线性的；AEx-WELM 和 Adaboost-WELM 虽然训练时间比 ELM 用时长很多，训练时间的增长规模是和 Adaboost 迭代次数相关的，但是我们发现这个迭代次数并不是随着文本向量维度和隐藏节点增加而线性增加的，也就是说训练时间的增加从总体上看是低于线性增长的，所以在 Reuters52 数据集中会产生基于 Adaboost 的 4 种 ELM 方法和 Voting-ELM 随着文本向量维度的增加越来越接近的现象。在 AEx-WELM 的 3 种方法 AE1-WELM、AE2-WELM、AE3-WELM 中，在训练时间和测试时间上基本一致，没有太大的差别，没有任何一种方法有明显的优势。

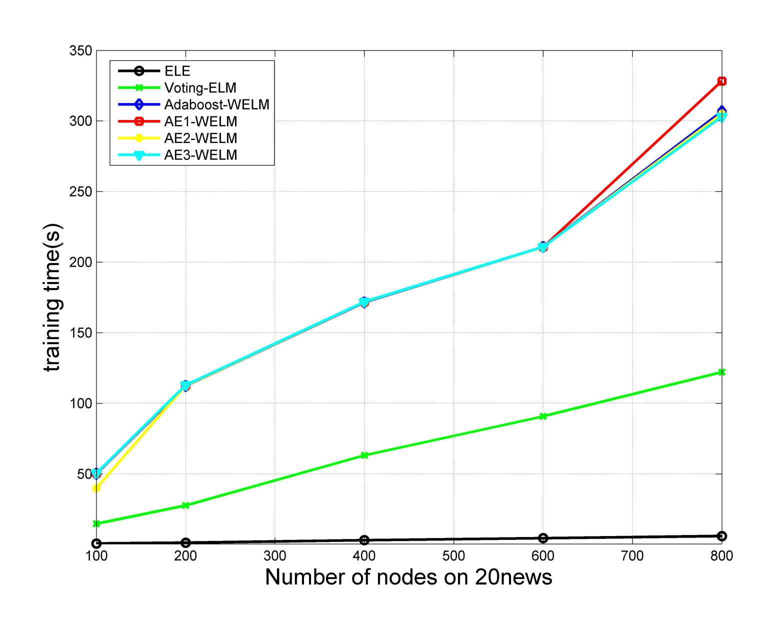


图 7 20newsgroups 数据集上的训练时间对比

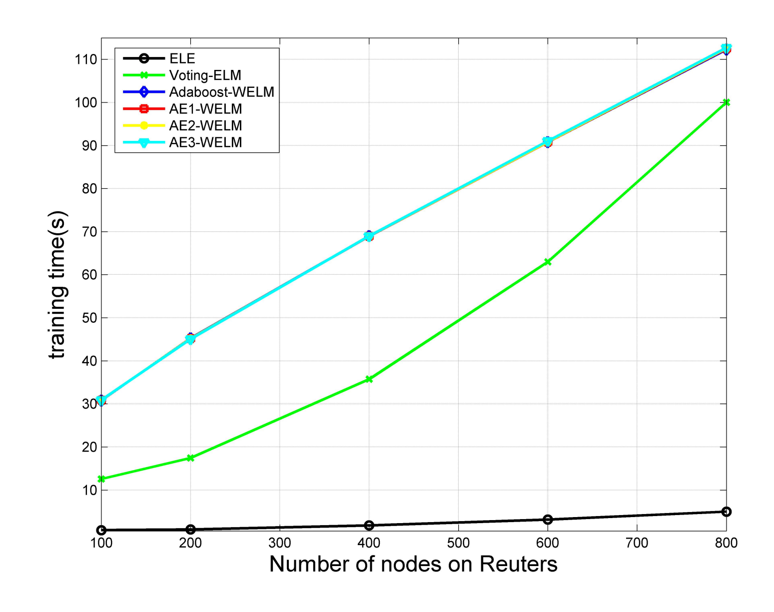


图 8 Reuters52 数据集上的训练时间对比

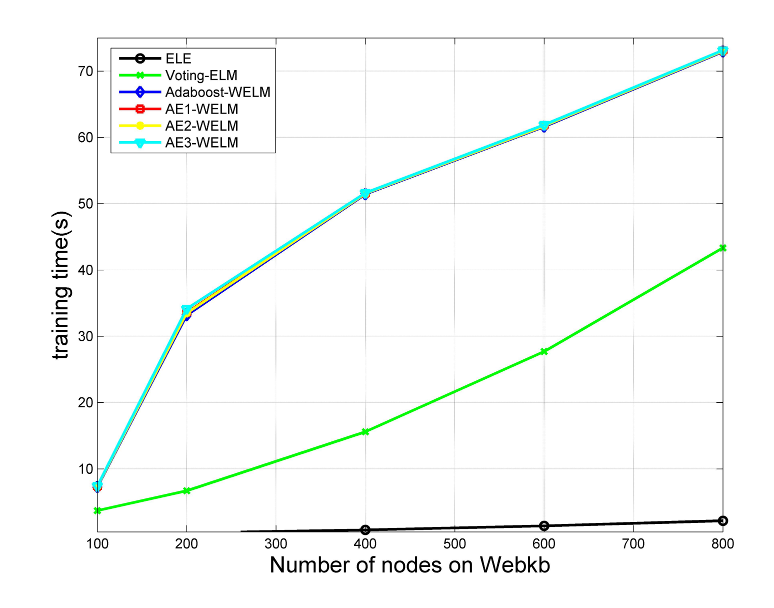


图 9 Webkb 数据集上的训练时间对比

5 结论

传统的 VSM 文本表达产生高维而稀疏的文本特征给极限学习机的计算增加了负担，本文针对这个问题将词向量模型作为文本表达方法。本文通过文本类别信息熵对样本的重要性进行度量，并且从样本重要性角度生成代价敏感矩阵和代价敏感因子，通过把代价敏感极限学习机集成到 Adaboost.M1 框架中以期提高文本分类性能。实验表明：在非平衡数据集和平衡数据集，AE1-WELM 的 AE3-WELM 综合分类性能指标均优于其他类型的极限学习机，其中 AE3-WELM 的整体性能优于 AE1-WELM。在将来的工作中，将研究如何在词向量的基础上进一步降低文本特征维度，选取更为合理的代价敏感函数来进一步减少 AEx-WELM 的计算花销以及如何进一步优化 AEx-WELM 框架，以获得更好的文本分类性能。

参考文献：

[1] Samworth R J. Optimal weighted nearest neighbour classifiers[J]. The Annals of Statistics, 2012, 40(5): 2733-2763.

[2] Zhang H, Berg A C, Maire M, et al. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition[C]. Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, 2006: 2126-2136.

[3] Chen J, Huang H, Tian S, et al. Feature selection for text classification with Naïve Bayes[J]. Expert Systems with Applications, 2009, 36(3): 5432-5435.

[4] Takahashi F, Abe S. Decision-tree-based multiclass support vector machines[C]. Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on, 2002: 1418-1422.

[5] Liu W, Song N. A fuzzy approach to classification of text documents[J]. Journal of Computer Science and Technology, 2003, 18(5): 640-647.
[6] Widyantoro D H, Yen J. A fuzzy similarity approach in text classification task[C]. Fuzzy Systems, 2000. FUZZ IEEE 2000. The Ninth IEEE International Conference on, 2000: 653-658.
[7] Chang C-C, Lin C-J. LIBSVM: a library for support vector machines[J]. ACM transactions on intelligent systems and technology (TIST), 2011, 2(3): 27.
[8] Ghiassi M, Olschimke M, Moon B, et al. Automated text classification using a dynamic artificial neural network model[J]. Expert Systems with Applications, 2012, 39(12): 10967-10976.
[9] Lam H-K, Ekong U, Liu H, et al. A study of neural-network-based classifiers for material classification[J]. Neurocomputing, 2014, 144: 367-377.
[10] Bigi B. Using Kullback-Leibler distance for text categorization[C]. European Conference on Information Retrieval, 2003: 305-319.
[11] Huang G-B, Zhu Q-Y, Siew C-K. Extreme learning machine: theory and applications[J]. Neurocomputing, 2006, 70(1): 489-501.
[12] Qin A K, Huang V L, Suganthan P N. Differential evolution algorithm with strategy adaptation for global numerical optimization[J]. IEEE transactions on Evolutionary Computation, 2009, 13(2): 398-417.
[13] Liu Y, Loh H, Tor S. Comparison of extreme learning machine with support vector machine for text classification[J]. Innovations in Applied Artificial Intelligence, 2005: 390-399.
[14] Zheng W, Qian Y, Lu H. Text categorization based on regularization extreme learning machine[J]. Neural Computing and Applications, 2013, 22(3-4): 447-456.
[15] Zheng W, Tang H, Qian Y. Collaborative work with linear classifier and extreme learning machine for fast text categorization[J]. World Wide Web, 2015, 18(2): 235-252.
[16] Zhao X-G, Wang G, Bi X, et al. XML document classification based on ELM[J]. Neurocomputing, 2011, 74(16): 2444-2451.
[17] Zhao X, Bi X, Qiao B. Probability based voting extreme learning machine for multiclass XML documents classification[J]. World Wide Web, 2014, 17(5): 1217-1231.
[18] Duan L, Yuan B, Wu C, et al.: Text-image separation and indexing in historic patent document image based on extreme learning machine, Proceedings of ELM-2014 Volume 2: Springer, 2015: 299-307.
[19] Yu H, Chen L, Zheng W. Chinese text sentiment classification based on kernel extreme learning machines[J]. Journal of China University of Metrology, 2016, 2: 020.
[20] 李永强. 基于粒子群优化的极限学习机的 XML 文档分类中的研究与应用[D]. 东北大学, 2013.
Li Yongqiang, Research and Application of XML Classification Based on Extreme learning Mchine with Particle Swarm Optimization[D]. Northeastern University, 2013
[21] Roul R K, Nanda A, Patel V, et al. Extreme learning machines in the field of text classification[C]. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015 16th IEEE/ACIS International Conference on, 2015: 1-7.
[22] Roul R K, Sahay S K. K-means and Wordnet Based Feature Selection Combined with Extreme Learning Machines for Text Classification[C]. International Conference on Distributed Computing and Internet Technology, 2016: 103-112.
[23] 陆慧娟, 安春霖, 马小平,等. 基于输出不一致测度的极限学习机集成的基因表达数据分类[J]. 计算机学报. 2013, (2): 341-348.

LU Hui-Juan, An Chun-Lin, Ma Xiao-Ping, et al. Disagreement measure Based Ensemble of Extreme learning Machine for Gene Expression Data Classification[J]. Chinanese Journal of Computers. 2013, (2): 341-348.
[24] Ditterrich T. Machine learning research: four current direction[J]. Artificial Intelligence Magzine, 1997, 4: 97-136.
[25] Jiang Y, Shen Y, Liu Y, et al. Multiclass AdaBoost ELM and its application in LBP based face recognition[J]. Mathematical Problems in Engineering, 2015, 2015.
[26] 黄海波, 李人宪, 黄晓蓉, 等. 基于样本熵与 ELM-Adaboost 的悬架减振器异响声品质预测[J]. 振动与冲击, 2016, (13): 125-133.
HUANG Hai-bo, LI Ren-xian, HUANG Xiao-rong, et al. Prediction of a suspension shock absorber's sound metric based on sample entropy and ELM-adaboost[J]. Journal of Vibration and Shock, 2016, (13): 125-133.
[27] Xu Y, Wang Q, Wei Z, et al. Traffic sign recognition based on weighted ELM and AdaBoost[J]. Electronics Letters, 2016, 52(24): 1988-1990.
[28] Li K, Kong X, Lu Z, et al. Boosting weighted ELM for imbalanced learning[J]. Neurocomputing, 2014, 128: 15-21.
[29] Freund Y, Schapire R E. A desicion-theoretic generalization of on-line learning and an application to boosting[C]. European conference on computational learning theory, 1995: 23-37.
[30] Turian J, Ratinov L, Bengio Y. Word representations: a simple and general method for semi-supervised learning[C]. Proceedings of the 48th annual meeting of the association for computational linguistics, 2010: 384-394.
[31] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.
[32] Sebastiani F. Machine learning in automated text categorization[J]. ACM computing surveys (CSUR), 2002, 34(1): 1-47.

(通讯作者: 李明 E-mail:liming@magicalthink.com)

作者贡献声明：

李明：提出研究思路，设计研究方案，进行实验，论文起草；
肖培伦：数据的获取、提供与分析，进行实验；
顾心盟：进行实验；
张矩：论文最终版本修订。